Publisher's Note

Salem Press is pleased to add *Principles of Computer Science* as the fourth title in a new *Principles of* series that includes *Chemistry, Physics, Astronomy,* and *Computer Science*. This new resource introduces students and researchers to the fundamentals of computer science using easy-to-understand language, giving readers a solid start and deeper understanding and appreciation of this complex subject.

The 117 entries range from 3D printing to Workplace Monitoring and are arranged in an A to Z order, making it easy to find the topic of interest. Entries include the following:

- Related fields of study to illustrate the connections between the various branches of computer science including computer engineering, software engineering, biotechnology, security, robotics, gaming, and programming languages;
- A brief, concrete summary of the topic and how the entry is organized;
- Principal terms that are fundamental to the discussion and to understanding the concepts presented;
- Illustrations that clarify difficult concepts via models, diagrams, and charts of such key topics as wide area networks (WAN), electronic circuits, and quantum computing;
- Photographs of significant contributors to the field of computer science;
- Sample problems that further demonstrate the concept presented;
- Further reading lists that relate to the entry.

This reference work begins with a comprehensive introduction to the field, written by editor Donald R. Franceschetti, PhD, starting with the development of the first computers, an explanation of analog versus digital computing, neural networks, the contributions of Alan Turing, and a discussion of how advances from vacuum tubes to digital processors have fundamentally changed the way we live.

The book's backmatter is another valuable resource and includes:

- A timeline of the developments that related to and led up to the first modern computer, starting in 2400 BCE with the invention of the abacus in Babylonia and ending with the 1949 *Popular Mechanics* prediction that computers of the future might weigh no more than 1.5 tons;
- A time of the development of microprocessors from the Intel 4004 in 1971 to the IBM POWER8 in 2014;
- Nobel Notes that explain the significance of the prizes in physics to the study of the science and
- List of important figures in computer science and their key accomplishment;
- Glossary;
- General bibliography; and
- Subject index.

Salem Press and Grey House Publishing extend their appreciation to all involved in the development and production of this work. The entries have been written by experts in the field. Their names and affiliations follow the Editor's Introduction.

Principles of Computer Science, as well as all Salem Press reference books, is available in print and as an e-book. Please visit www.salempress.com for more information.

Editor's Introduction

What is Computer Science?
Computer science generally refers to the body of knowledge that allows humans to use mechanical aids to do mathematical calculations very efficiently. While the term computer originally referred to a human who was able to do simple arithmetic quickly and accurately, the name has been transferred to programmable automata, which are able to do the same thousands to billions of times faster.

Analogue or Digital
There are two basic kinds of computers: analogue and digital. In an analog process, the system of interest is modeled by a system obeying the same dynamical laws. Thus a system of mechanical vibrations might be modeled by a system of capacitors, inductors, and resistors. In a digital computer, the quantity of interest is generally encoded in some way and one obtains coded representation of the solution. Some of the advances are fairly mundane and are actually possible to achieve with low-tech adding machines, if one has enough time. Others are a bit more esoteric, requiring some abstract mathematics. Almost all computing machines, with the notable exceptions of the abacus and slide rule, have come into existence since the beginning of the twentieth century and rest on the fundamental discoveries made about electromagnetism since 1800, as well as more recent insights gained into the process of computation.

Digital Computation Requires a Code
Digital computation begins with the selection of a code so that information can be stored and sent over significant distances. It can be difficult for those living in the twenty-first century to appreciate the full impact of the development of telegraphy, by which information (letters and numbers) could be transmitted from one place to another as a series of dots and dashes, using a pair of telegraph wires strung between two cities along with the necessary relay equipment. For the first time, the results of a baseball game or a battle or the day's stock trading could be known almost instantly. It's not surprising that so many innovators in the area of information exchange began their careers as telegraph operators.

The earliest innovations in telegraphy greatly increased the number of messages that could be sent simultaneously. It was eventually learned that by using more than two conductors, several symbols could be sent at the same time. Of course, it should be noted that the original Morse code used to tap out messages with a series of dots and dashes was highly inefficient and error prone. Suppose seven wires carried voltage simultaneously. With seven voltages, one can encode 2^7 =128 symbol choices, a calculation that accounts for the digits of the decimal system plus upper and lower case letters plus several punctuation marks. The telegraph thus becomes a teletype machine capable of transmitting information a hundred times faster.

Once the process of telegraphy was in place, it was a simple matter to switch from the network of telegraph stations to radio. Once amplitude modulation was discovered voice could be transmitted over the air or over wires with ease. Television eventually followed. Many of the pioneers of electronic technology were self-educated men—men like Thomas Edison, and Michael Faraday before him—who could read well enough but functioned outside the establishment of colleges and academic degrees. Even today there is some controversy over the value of traditional education as many pioneers of the computer sciences dropped their pursuit of traditional degrees, such as Bill Gates and Steve Jobs, while means that the question still remains open.

Still, formal education came to be an important aspect of the inexorable march toward the computer age. In the United Kingdom, while social standing may have prevent many from attending a university enrollment, entire institutions were set up to teach those who had not the pedigree to be accepted at Oxford or Cambridge, notably the Royal Institution, a school operating under Royal charter and featuring both Sir Humphrey Davy and Michael Faraday (who both declined to be knighted) as lecturers. Things were somewhat more democratic in the United States, with public universities in almost every state that were supported by agriculture and expected to make a contribution in return. Private institutions set up chapters of Phi Beta Kappa to encourage traditional study of

Latin and Greek, while Institutes of Technology like the Massachusetts Institute of Technology offered degrees in the Sciences and Engineering.

One of the important areas of computer science is the selection and security of codes. An in-depth treatment of the topic is provided by Claude Shannon and Warren Weaver in two papers originally published in *Scientific American* and in the *Bell System Technical Journal*. The very existence of a widely respected scientific journal published not by a non-profit organization, but rather as a house organ of Bell System shows how the system of scientific publication had to adapt to the emergence of a very large corporation that of necessity has control of a major portion of the work to be done.

Shannon begins by defining the entropy of a text in terms of the probabilities of the symbols. Scrambling the symbols in a text corresponds to an increase in entropy and the probabilities of certain errors. Shannon and others have made very effective use of the entropy concept, which was originally proposed by Ludwig Boltzmann in the context of the molecular theory of heat. Among other things, the creation of information results in the generation of heat and so one of the major technical issues in the design of highly powerful computers is the need to transport the heat away as it is generated.

The Development of the Digital Computer

In 1900, the great German mathematician David Hilbert gave a talk at the Second International Conference of Mathematicians in Paris, France, during which he described twenty-three mathematical problems that he expected to be solved in the twentieth century. His tenth problem dealt with polynomial equations with integer coefficients (Diophantine equations). Hilbert asked if it could be determined in a finite number of steps whether the equation could be solved in rational numbers. The matter in question was not to find the solution, but simply to find out if a solution could be found.

One of the world's true polymaths, Alan Turing, took on the challenge of solving Hilbert's problem–the so-called "decision problem" or *Entscheidungsproblem*. To solve this problem, Turing looked closely at the process of solution. He begins by looking at a solution as a practical problem, in other words, as the sequence of steps that a mathematician or a young child might go through solving an arithmetic problem by working out the solution one digit at a time. It is a truism that the solution to practical problems is sometimes found by "thinking outside the box" and considering mechanisms that might seem to have nothing to do with the problem at hand. In the process of solving Hilbert's decision problem, Turing conceived of a machine that could read and write one digit at a time, based on instructions that could be written on the same tape that would carry the solution—known today as the Turing Machine.

It has been suggested by a number of Turing biographers that since Alan enjoyed working with actual typewriters, it was in fact his practical bent that led him to conceptualize the Turing machine and then the universal Turing machine. The set of instructions constitutes the computer program. The numbers written on the tape at the start of the tape constitute the *input* or *data* and the numbers written on the tape after the computer is done are the *output*. Numbers which might be written by such a machine are called computable numbers. Turing published his first paper "On Computable Numbers with an application to the *Entscheidungsproblem*" in 1937. This paper marks the theoretical development of the digital computer.

Turing received his PhD in mathematics in 1938 and by 1939 was at work as a cryptographer in the British Foreign Office. It was during this period of his life—which became the subject of the Academy Award-winning movie, *The Imitation Game*—that he was able to apply his efforts to produce the "enigma" coding machine capable of breaking the German code and ultimately, saving numerous lives.

It turns out that one can write a program capable of accepting the description of another Turing machine as data and then emulating it. Such programs are called universal Turing machines. The modern programmable digital computer is, from an abstract point of view, just such a machine. As one of the first individuals to take the possibility of artificial intelligence seriously, Turing proposed a test, known today as the Turing test, to tell whether a machine could be considered intelligent.

Background of the Turing Machine

The digital computer has undergone many changes since Turing first developed it to deal with the urgencies of the Second World War. Today's digital

computer as a practical device is the result of developments in a large number of fields.

The nineteenth century had been a trying time for the natural sciences. While the physics of 1900 left little room for new discoveries, the discovery of subatomic particles like the electron and the nuclear structure of the atom called for a fundamental reexamination of assumptions not challenged since the time of Isaac Newton. Einstein had opened the possibility that the geometry of space was not that proposed by Euclid over two thousand years ago. Philosophically-minded scientists turned to mathematics for sure and certain knowledge. Of all areas of mathematics, the most noncontroversial was the theory of arithmetic, in which there was only one item of unfinished business: the principle of mathematical induction, which seemed to be an unnecessary part of the structure of mathematics. This principle stated that if N was an integer, and if P(N) was a true statement about N that implied the truth of the statement P(N+1), then P(N) was true for all N. A number of philosophers of mathematics went to work trying to show that mathematics could be constructed in a self- consistent way without the troublesome postulate.

By far the most thoroughgoing approach to eliminating the principle of mathematical induction could be found the three-volume *Principia Mathematica,* published in 1911 (and still in print!) by Bertrand Russell and Alfred North Whitehead. Although a fallacy inherent in *Principia Mathematica* was soon discovered, the book remains a valuable exposition of symbolic logic. In 1931 Kurt Godel ,who would later become a very close friend of Albert Einstein published a paper "On formally undecidable systems…" that put an end to the matter by showing that in any system of axioms for arithmetic that allowed multiplication, there would necessarily be propositions which could not be decided within the system.

Neural Networks

In 1943, however, Warren S. McCulloch and Walter Pitts published "A logical calculus of the ideas immanent in nervous activity" showing that for each formula in the notation of *Principia Mathematica*, an equivalent network of all or none neurons could be found. Work on neural networks has continued to this day.

Von Neumann's Contributions

Von Neumann was a very unusual physicist, one of a group of immigrants known collectively as "Martians" because their Hungarian speech resembled no known European language; other Martians included Edward Teller and Leo Szilard. Unlike the majority of academics, he had adequate financial means and was able to attend classes at several universities. He did not shy away from positions of political and social influence, even accepting President Eisenhower's appointment as Commissioner of Atomic Energy. While his PhD degree was given for a thesis on the "Mathematical Foundations of Quantum Mechanics," he was also interested in practical electronics. As a member of the Institute for Advanced Studies at Princeton New Jersey, he oversaw the construction of one of the first vacuum tube computers. In the last years of his life, Dr. John von Neumann prepared to give the Silliman lectures at Yale University, his topic: The Computer and the Brain. While he was unable to deliver the Silliman lectures for health reasons, the manuscript was eventually published,

The First Electronic Computers

Prepared with a basic understanding of neural networks as devices that model human thought, we turn now to the problems of electronic devices that can undertake computation. The essential requirement is for a two-state (or greater) device. The Edison effect, observed by Thomas Edison in 1883, provided the basis for the first such devices. Edison had noted the emission of electrons from a heated metal electrode when maintained in vacuum, but he merely noted the effect in his laboratory notebook. Eventually, however, the British engineer, Sir John Ambrose Fleming, used a heated cathode in the first vacuum tube diode, also called a rectifier since it only allowed electron flow from the heated cathode. An American inventor, Lee de Forest, added a third element, or grid, that allowed a small voltage to control a much larger current.

The Vacuum Tube Era

The first electronic computers employed vacuum tube circuitry and were not always dependable. Vacuum tubes contained a heated element, or filament, that was essential to its proper function. Once the filament burned out, the tube would not conduct when it should. There were two possible approaches

to solving the problem. One could design a test calculation that would use every tube in the computer and for which the result is known. If one ran the test calculation before and after a new calculation was run and obtained the same results, one could be relatively sure that no tubes had burned out during the calculation. An alternative suggested by von Neumann was to build redundancy into the circuit so that the results obtained could be trusted even if a few components proved unreliable.

The Transistor Era: Computers become Profitable

While there were fortunes to be made making business machines, the computer industry offered new opportunities to creative individuals. As semiconductor devices replaced vacuum tubes, it became clear that a few creative individuals could compete effectively against older and well-established corporations. While there was still a need for large mainframe computers, the personal computer had the virtues of both versatility and low cost. The leaders in the field became the hobbyists, many of them still in high school, who established Microsoft and Apple Computer. Bill Gates, founder of Microsoft, would become one of the world's richest men. Steve Jobs, cofounder of Apple Computer, brought the standalone microcomputer to new levels of competitiveness. Facebook's evolution created an increasing appetite for social media. New mechanisms of marketing were found. The internet, first started in order to connect a few physics research laboratories, became a "place" where all manner of goods and services could be bought and sold.

The Future

Clearly computer science is in the midst of a revolution of its own making. It is not possible to predict where or when things will "settle down," or if they ever actually will. Nonetheless, a few outcomes are already clear: Information will become more and more available and there will be an increased need for professionals in the great many fields impacted by the advances described here. If this volume helps the reader to see even a bit more clearly through the maze of opportunity, it has been well worth the effort to bring it together.

—Donald Franceschetti, PhD

Bodanis, David. *Electric Universe: The Shocking True Story of Electricity.* New York: Crown Publishers, 2005. Print.

Penrose, Roger. *The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics.* Oxford: Oxford University Press, 1989. Print.

Shannon, Claude E, and Warren Weaver. *The Mathematical Theory of Communication.* Urbana: University of Illinois Press, 1999. Print.

Yandell, Ben. The Honors Class: Hilbert's Problems and Their Solvers. Natick, Mass: A.K. Peters, 2002. Print.

Von, Neumann J, and Ray Kurzweil. *The Computer & the Brain.* New Haven, Conn.; London : Yale University Press, 2012. Print.

A

AGILE ROBOTICS

FIELDS OF STUDY

Robotics

ABSTRACT

Movement poses a challenge for robot design. Wheels are relatively easy to use but are severely limited in their ability to navigate rough terrain. Agile robotics seeks to mimic animals' biomechanical design to achieve dexterity and expand robots' usefulness in various environments.

PRINICIPAL TERMS

- **anthropomorphic:** resembling a human in shape or behavior; from the Greek words *anthropos* (human) and *morphe* (form).
- **autonomous:** able to operate independently, without external control.
- **biomechanics:** the study of how living things move and the laws governing their movement.
- **dexterity:** finesse; skill at performing delicate or precise tasks.
- **dynamic balance:** the ability to maintain balance while in motion.
- **humanoid:** resembling a human.

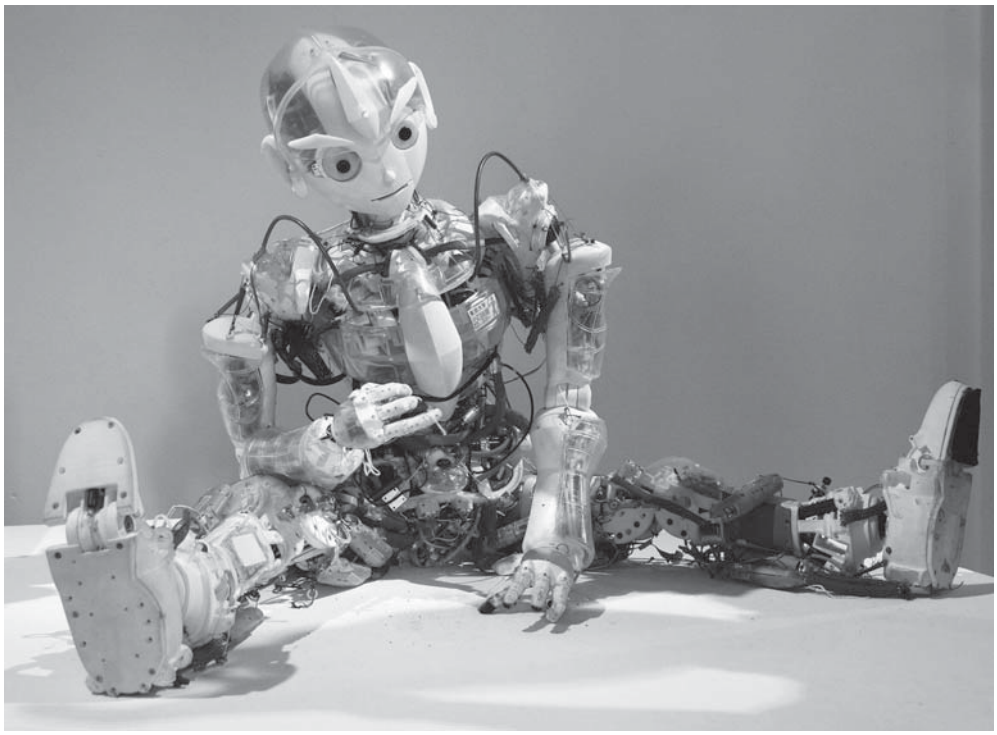
Robots That Can Walk

Developing robots that can match humans' and other animals' ability to navigate and manipulate their environment is a serious challenge for scientists and engineers. Wheels offer a relatively simple solution for many robot designs. However, they have severe limitations. A wheeled robot cannot navigate simple stairs, to say nothing of ladders, uneven terrain, or the aftermath of an earthquake. In such scenarios, legs are much more useful. Likewise, tools such as simple pincers are useful for gripping objects, but they do not approach the sophistication and adaptability of a human hand with opposable thumbs. The cross-disciplinary subfield devoted to creating robots that can match the dexterity of living things is known as "agile robotics."

Inspired by Biology

Agile robotics often takes inspiration from nature. Biomechanics is particularly useful in this respect, combining physics, biology, and chemistry to describe how the structures that make up living things work. For example, biomechanics would describe a running human in terms of how the human body—muscles, bones, circulation—interacts with forces such as gravity and momentum. Analyzing the activities of living beings in these terms allows roboticists to attempt to recreate these processes. This, in turn, often reveals new insights into biomechanics. Evolution has been shaping life for millions of years through a process of high-stakes trial-and-error. Although evolution's "goals" are not necessarily those of scientists and engineers, they often align remarkably well.

Boston Dynamics, a robotics company based in Cambridge, Massachusetts, has developed a prototype robot known as the Cheetah. This robot mimics the four-legged form of its namesake in an attempt to recreate its famous speed. The Cheetah has achieved a land speed of twenty-nine miles per hour—slower than a real cheetah, but faster than any other legged robot to date. Boston Dynamics has another four-legged robot, the LS3, which looks like a sturdy mule and was designed to carry heavy supplies over rough terrain inaccessible to wheeled transport. (The LS3 was designed for military use, but the project was shelved in December 2015 because it was too noisy.) Researchers at the Massachusetts Institute of Technology (MIT) have built a soft robotic fish. There are robots in varying stages of development



Agile robots are designed to have dexterity, flexibility, and a wider range of motions to allow for more responsiveness to their surroundings. By Manfred Werner - Tsui, CC-BY-SA-3.0 (http://creativecommons.org/licenses/by-sa/3.0/), via Wikimedia Commons.

that mimic snakes' slithering motion or caterpillars' soft-bodied flexibility, to better access cramped spaces.

In nature, such designs help creatures succeed in their niches. Cheetahs are effective hunters because of their extreme speed. Caterpillars' flexibility and strength allow them to climb through a complex world of leaves and branches. Those same traits could be incredibly useful in a disaster situation. A small, autonomous robot that moved like a caterpillar could maneuver through rubble to locate survivors without the need for a human to steer it.

Humanoid Robots in a Human World

Humans do not always compare favorably to other animals when it comes to physical challenges. Primates are often much better climbers. Bears are much stronger, cheetahs much faster. Why design anthropomorphic robots if the human body is, in physical terms, relatively unimpressive?

NASA has developed two different robots, Robonauts 1 and 2, that look much like a person in a space suit. This is no accident. The Robonaut is designed to fulfill the same roles as a flesh-and-blood astronaut, particularly for jobs that are too dangerous or dull for humans. Its most remarkable feature is its hands. They are close enough in design and ability to human hands that it can use tools designed for human hands without special modifications.

Consider the weakness of wheels in dealing with stairs. Stairs are a very common feature in the houses and communities that humans have built for themselves. A robot meant to integrate into human society could get around much more easily if it shared a similar body plan. Another reason to create humanoid robots is psychological. Robots that appear more human will be more accepted in health care, customer service, or other jobs that traditionally require human interaction.

Perhaps the hardest part of designing robots that can copy humans' ability to walk on two legs is achieving dynamic balance. To walk on two legs, one must adjust one's balance in real time in response to each step taken. For four-legged robots, this is less of an issue. However, a two-legged robot needs sophisticated sensors and processing power to detect and respond quickly to its own shifting mass. Without this, bipedal robots tend to walk slowly and awkwardly, if they can remain upright at all.

The Future of Agile Robotics

As scientists and engineers work out the major challenges of agile robotics, the array of tasks that can be given to robots will increase markedly. Instead of being limited to tires, treads, or tracks, robots will navigate their environments with the coordination and agility of living beings. They will prove invaluable not just in daily human environments but also in more specialized situations, such as cramped-space disaster relief or expeditions into rugged terrain.

—Kenrick Vezina, MS

Bibliography

Bibby, Joe. "Robonaut: Home." *Robonaut.* NASA, 31 May 2013. Web. 21 Jan. 2016.

Gibbs, Samuel. "Google's Massive Humanoid Robot Can Now Walk and Move without Wires." *Guardian.* Guardian News and Media, 21 Jan. 2015. Web. 21 Jan. 2016.

Murphy, Michael P., and Metin Sitti. "Waalbot: Agile Climbing with Synthetic Fibrillar Dry Adhesives." *2009 IEEE International Conference on Robotics and Automation.* Piscataway: IEEE, 2009. *IEEE Xplore.* Web. 21 Jan. 2016.

Sabbatini, Renato M. E. "Imitation of Life: A History of the First Robots." *Brain & Mind* 9 (1999): n. pag. Web. 21 Jan. 2016.

Schwartz, John. "In the Lab: Robots That Slink and Squirm." *New York Times.* New York Times, 27 Mar. 2007. Web. 21 Jan. 2016.

Wieber, Pierre-Brice, Russ Tedrake, and Scott Kuindersma. "Modeling and Control of Legged Robots." *Handbook of Robotics.* Ed. Bruno Siciliano and Oussama Khatib. 2nd ed. N.p.: Springer, n.d. (forthcoming). *Scott Kuindersma—Harvard University.* Web. 6 Jan. 2016

ALGOL

FIELDS OF STUDY

Programming Languages

ABSTRACT

The ALGOL programming language was developed in 1958 as a program for the display of algorithms. It was elegant and included several design features that have since become staple features of advanced programming languages. ALGOL programs and procedures employ a head-body format, and procedures can be nested within other procedures. ALGOL was the first programming language to make use of start-end delimiters for processes within procedures, a feature now common in advanced object-oriented programming languages. The language allowed recursion and iterative procedures, dynamic array structures, and user-defined data types. Despite its elegance and advanced features, however, ALGOL never became widely used and is now one of the oldest and least used of programming languages.

PRINICIPAL TERMS

- **algorithm:** a set of step-by-step instructions for performing computations.
- **character:** a unit of information that represents a single letter, number, punctuation mark, blank space, or other symbol used in written language.
- **function:** instructions read by a computer's processor to execute specific events or operations.
- **object-oriented programming:** a type of programming in which the source code is organized into objects, which are elements with a unique identity that have a defined set of attributes and behaviors.

- **main loop:** the overarching process being carried out by a computer program, which may then invoke subprocesses.
- **programming languages:** sets of terms and rules of syntax used by computer programmers to create instructions for computers to follow. This code is then compiled into binary instructions for a computer to execute.
- **syntax:** rules that describe how to correctly structure the symbols that comprise a language.

History and Development of ALGOL

The name ALGOL is a contraction from ALGOrithmic Language. It was developed by a committee of American and European computer scientists at ETH, in Zurich, Switzerland, as a language for the display of algorithms. The detailed specifications for the language were first reported in 1960, as ALGOL-58, and there have been several revisions and variations of the language since its inception. These include the updated ALGOL-60, ALGOL-N, ALGOL-68, ALGOL-W and Burroughs Extended ALGOL. The language was developed from the FORTRAN language, which appeared in 1956. Both ALGOL and FORTRAN strongly influenced the development of later languages such as BASIC, PL/1 and PL/C, Euler and Pascal. Many ALGOL programmers agree that the development of ALGOL-68 made the language much more complex and less elegant than the previous version ALGOL-60, ultimately leading to its fall into disfavor with computer users. It is now one of the oldest and least used of all programming languages.

Program Characteristics

The syntax of ALGOL is rather logical, using natural-language reserved keywords such as comment, begin and end and the ":" (colon) character to identify standard arithmetical operators. The ALGOL program format utilizes a two-part "block" structure in its main loop that has since become a familiar feature of most modern computer languages such as C/C++, Java and many others. The first block, called the "head," consists of a series of type declarations similar to those used in FORTRAN type declaration statements. The declarations specify the class of the entities used in the program, such as integers, real numbers, arrays, and so on. For example, the head section of an ALGOL-60 program may be

```
comment An ALGOL-60 sorting program
procedure Sort (Y, Z)
        value Z;
        integer Z; real array Y;
```

As can be seen, comment lines for documentation are identified by the keyword comment, and procedures are identified by the keyword procedure followed by the name of the procedure. The second block, which is the main part of the program, consists of a sequence of statements that are to be executed. It is initiated by the keyword begin and terminated by the keyword end. In fact this is the common structure of ALGOL procedures, and the program format supports procedures within procedures. For example, the following body section of a sorting procedure contains a second procedure within it, as

```
begin
        real x;
        integer i, j;
        for i := 2 until Z do begin
                x := Y[i];
                for j := i-1 step -1 until 1 do
                        if x >= A[j] then begin
                                A[j+1] := x; goto Found
                        end else
                                A]j+1] := A[j];
                A[1] := x;
        Found:
                end
        end
end Sort
```

Combining these two code segments produces an ALGOL-60 program that sorts a series of numbers. Examination of this body code segment reveals that the begin statement is followed by the head of a nested procedure, which is in turn followed by the body statements of the secondary procedure before the terminating end statements. The ability to support nested procedures foreshadowed the concept of a function in the context of object-oriented programming. It is a vitally important aspect of computer programming, and has proven the robustness of the

form in many languages since the development of ALGOL.

SAMPLE PROBLEM

Comment the following section of code:

```
begin
        real x;
        integer i, j;
        for i := 2 until Z do begin
                x := Y[i];
```

Answer:

```
begin
comment identify the variable types as real
numbers and integers
        real x;
        integer i, j;
        comment an iterative routine to as-
        sign elements in the array
        for i := 2 until Z do begin
                comment assigns the value of x
                to the array element
                x := Y[i];
```

The ALGOL Legacy

While the ALGOL programming language was overshadowed by the FORTRAN programming language and did not become popular, it featured a number of innovative aspects that have since become staples of essentially all major programming languages. In particular, the nested procedure structure foreshadowed the object-oriented programming style. It was the first language to make use of start-end identifiers as block delimiters, a convention that has carried over in object-oriented languages using function delimiters such as the { and } brace characters. ALGOL procedures included conditional statements (if...then and if...else), and iterative statements (for... until). Functions could call themselves in recursive computations (as in the statement i = i + j), and ALGOL enabled dynamic arrays in which the subscript range of the array elements was defined by the value of a variable, as in the array element i_j. ALGOL also used reserved keywords that could not be used as identifiers by a programmer, but it also allowed user-defined data types.

—Richard M. Renneboog M.Sc.

Bibliography

Malcolme-Lawes, D.J. (1969) *Programming – ALGOL* London, UK: Pergamon Press. Print.

Organick, E.I., Forsythe, A.I. and Plummer, R.P. (1978) *Programming Language Structures* New York, NY: Academic Press. Print.

O'Regan, Gerard (2012) *A Brief History of Computing* 2nd ed., New York, NY: Springer-Verlag. Print.

Wexelblat, Richard L. (1981) *History of Programming L:anguages* New York, NY: Academic Press. Print.

Rutishauer, Heinz (1967) "Description of ALGOL-60" Chapter in Bauer, F.L., Householder, I.S., Olver, F.W.J., Rutishauer, H., Samelson, K. and Stiefel, E., eds. *Handbook for Automatic Computation* Berlin, GER: Springer-Verlag. Print.

HISTORY OF EVENTS LEADING UP TO THE DEVELOPMENT OF MODERN COMPUTERS

circa 2400 BCE
The abacus – the first known calculator, was probably invented by the Babylonians as an aid to simple arithmetic around this time period.

circa 1115 BCE
The south-pointing chariot was invented in ancient China. It was the first known geared mechanism to use a differential gear.

circa 500 BCE
First known use of 0 (by mathematicians in ancient India around this date.

circa 500 BCE
Indian grammarian Pāṇini formulated the grammar of Sanskrit (in 3959 rules) known as the Ashtadhyayi. His grammar had the computing power equivalent to a Turing machine. The Panini-Backus form used to describe most modern programming languages is also significantly similar to Pāṇini's grammar rules.

circa 300 BCE
Indian mathematician/scholar/musician Pingala first described the binary number system which is now used in the design of essentially all modern computing equipment.

circa 200 BCE
The Chinese invented the suanpan (Chinese abacus) which was widely used until the invention of the modern calculator, and continues to be used in some cultures today.

circa 125 BCE
The Antikythera mechanism: A clockwork, analog computer believed to have been designed and built in the Corinthian colony of Syracuse. The mechanism contained a differential gear and was capable of tracking the relative positions of all then-known heavenly bodies.

circa 100 BCE
Chinese mathematicians first used negative numbers.

circa 60 BCE
Heron of Alexandria made numerous inventions, including "Sequence Control." This was, essentially, the first computer program. He also made numerous innovations in the field of automata, which are important steps in the development of robotics.

circa 200
Jaina mathematicians invented logarithms.

circa 600
Indian mathematician Brahmagupta was the first to describe the modern place-value numeral system.

724
Chinese inventor Liang Lingzan built the world's first fully mechanical clock; the earliest true computers, made a thousand years later, used technology based on that of clocks.

820
Persian mathematician, Muḥammad ibn Mūsā al-Khwārizmī, described the rudiments of modern algebra. The word "algorithm" is derived from al-Khwarizmi's Latinized name "Algoritmi".

circa 850
Arab mathematician, Al-Kindi (Alkindus gave the first known recorded explanation of cryptanalysis in "A Manuscript on Deciphering Cryptographic Messages". In particular, he is credited with developing the frequency analysis method whereby variations in the frequency of the occurrence of letters could be analyzed and exploited to break encryption ciphers.

850
The Banū Mūsā brothers, in their "Book of Ingenious Devices", invented an organ which played interchangeable cylinders automatically. They also invented an automatic flute player which appears to have been the first programmable machine.

TIMELINE OF MICROPROCESSORS

Year	Microprocessors
1971	Intel 4004
1972	Fairchild PPS-25; Intel 8008; Rockwell PPS-4
1973	Burroughs Mini-D; National IMP-16; NEC μCOM
1974	General Instrument CP1600; Intel 4040, 8080; Mostek 5065; Motorola 6800; National IMP-4, IMP-8, ISP-8A/500, PACE; Texas Instruments TMS 1000; Toshiba TLCS-12
1975	Fairchild F-8; Hewlett Packard BPC; Intersil 6100; MOS Technology 6502; RCA CDP 1801; Rockwell PPS-8; Signetics 2650
1976	RCA CDP 1802; Signetics 8x300; Texas Instruments TMS9900; Zilog Z-80
1977	Intel 8085
1978	Intel 8086; Motorola 6801, 6809
1979	Intel 8088; Motorola 68000; Zilog Z8000
1980	National Semi 16032; Intel 8087
1981	DEC T-11; Harris 6120; IBM ROMP
1982	Hewlett Packard FOCUS; Intel 80186, 80188,; 80286; Berkeley RISC-I
1983	Stanford MIPS; UC Berkeley RISC-II
1984	Motorola 68020; National Semi 32032; NEC V20
1985	DEC MicroVax II; Harris Novix; Intel 80386; MIPS R2000
1986	NEC V60; Sun SPARC; Zilog Z80000
1987	Acorn ARM2; DEC CVAX 78034; Hitachi Gmicro/200; Motorola 68030; NEC V70
1988	Intel 80386SX, i960; MIPS R3000
1989	DEC VAX DC520 Rigel; Intel 80486, i860
1990	IBM POWER1; Motorola 68040
1991	DEC NVAX; IBM RSC; MIPS R4000
1992	DEC Alpha 21064; Hewlett Packard PA-7100; Sun microSPARC I
1993	IBM POWER2, PowerPC 601; Intel Pentium

THE PIONEERS OF COMPUTER SCIENCE

al-Khwārizmī	The term "algorithm" is derived from the algorism, the technique of performing arithmetic with Hindu-Arabic numerals developed by al-Khwarizmi. Both "algorithm" and "algorism" are derived from the Latinized forms of al-Khwarizmi's name, Algoritmi and Algorismi, respectively.	c. 780 – c. 850
John Atanasoff	Built the first electronic digital computer, the Atanasoff–Berry Computer, though it was neither programmable nor Turing-complete (1939).	b. 1903 d. 1995
Charles Babbage	Originated the concept of a programmable general-purpose computer. Designed the Analytical Engine and built a prototype for a less powerful mechanical calculator.	1822 1837
John Warner Backus	Invented FORTRAN ("For"mula "Tran"slation), the first practical high-level programming language (1954), and he formulated the Backus–Naur form that described the formal language syntax (1963).	b. 1924 d. 2007
Jean Bartik	One of the first computer programmers, on ENIAC (1946). Worked with John Mauchly toward BINAC (1949), EDVAC (1949), UNIVAC (1951) to develop early "stored program" computers.	b. 1924 d. 2011
Sir Timothy John Berners-Lee	Invented worldwide web. With Robert Cailliau, sent first HTTP communication between client and server (1989).	b. 1955
George Boole	Formalized Boolean algebra (1854) in *The Laws of Thought*, the basis for digital logic and computer science.	b. 1815 d. 1864
Per Brinch Hansen	Developed the RC 4000 multiprogramming system introducing the concept of an operating system kernel and the separation of policy and mechanism (1967). Co-developed the monitor with Tony Hoare (1973), and created the first monitor implementation (1975). Implemented the first form of remote procedure call in the RC 4000 (1978)	1969
Nikolay Brusentsov	Built ternary computer Setun (1958).	b. 1925 d. 2014
Vannevar Bush	Originator of the Memex concept (1930), which led to the development of Hypertext.	b. 1890 d. 1974
David Caminer	Developed the LEO computer the first business computer with John Pinkerton, for J. Lyons and Co. (1951)	b. 1915 d. 2008